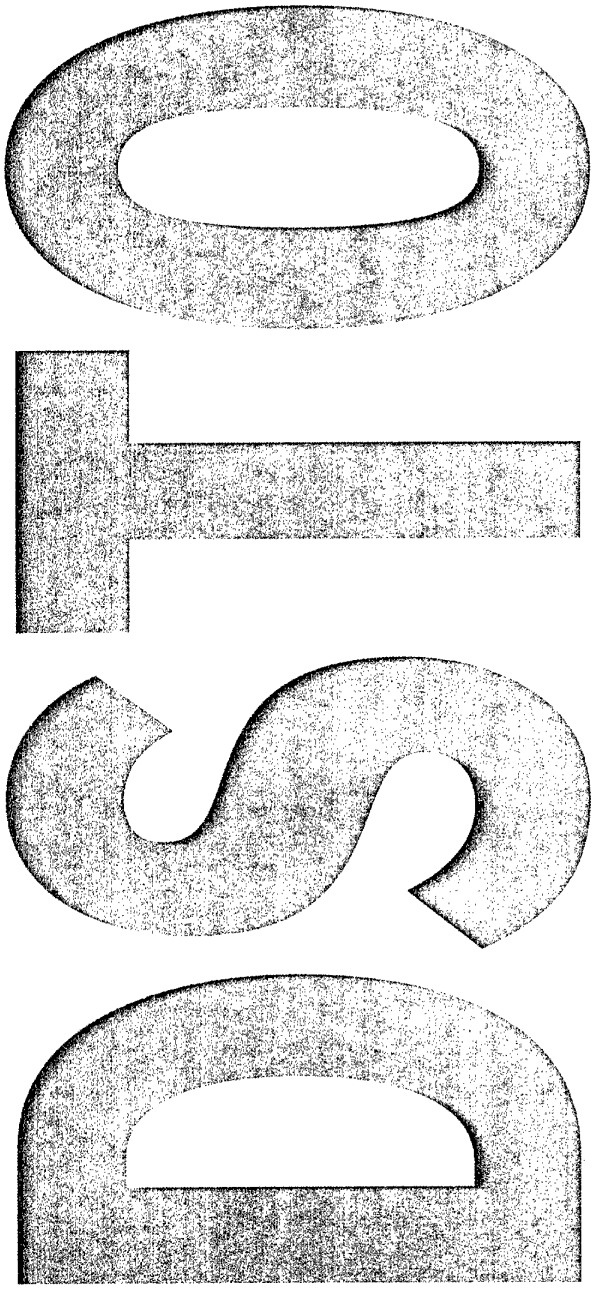
Australian Government
Department of Defence
Defence Science and
Technology Organisation

Sep 2003



Commercial-Off-The-Shelf (COTS) Systems, Architecture and Knowledge

Stephane Collignon

DSTO-TR-1493

Australian Government
Department of Defence
Defence Science and
Technology Organisation

Commercial-Off-The-Shelf (COTS) Systems, Architecture and Knowledge

Stephane Collignon

Electronic Warfare and Radar Division
Electronics and Surveillance Research Laboratory
DSTO-TR-1493

ABSTRACT

Large or specialised software systems-of-systems are now mostly composed of Commercial-Off-The-Shelf (COTS) software packages. In mainstream application areas, such as office automation, this approach has proved satisfactory once the teething problems have been overcome. Defence and Research have also followed this trend in a desire to cut costs and take advantage of the functionality of commercial software. However, the difference between the often demanding requirements for defence systems-of-systems, and those against which COTS software has been designed, has resulted in unfavourable outcomes. This paper discusses the insights that can be gleaned from systems theory and practice to improve the probability of successfully integrating COTS components into defence systems. In particular the paper will provide a practical definition of systems of systems (SOS) and then will compare the generic design drivers for software of this scale with that of modest stand alone packages. A systems approach is invoked to help understand the problem context presented by the SOS evolution task and uses this to identify ideas that may be able to mitigate some of the issues. The paper will conclude by identifying some practical approaches that can help realise evolving Defence and Research SOS.

Published by

DSTO Systems Sciences Laboratory
PO Box 1500
Edinburgh South Australia 5111

Telephone: (08) 8259 5555
Fax: (08) 8259 6567

AR-012-892
September 2003

APPROVED FOR PUBLIC RELEASE

Commercial-Off-The-Shelf (COTS) Systems, Architecture and Knowledge

EXECUTIVE SUMMARY

Today's software systems, which are built with Commercial-Off-The Shelf (COTS) hardware and software, are omnipresent. As these systems gradually replace the few remaining legacy systems, they merely substitute cost and performance problems with more complicated issues arising from the use of less controllable resources. Not only are COTS software products frequently utilised in environments that are completely at odds with those environments for which they were originally and specifically designed; but that they also come with limited documentation and a dearth of information about the design of the software products. This situation often leads to difficulties in the combined use of multiple COTS products in the development of a system-of-systems (SOS) architecture. Initially this paper addresses the definition of SOS, the players in the design and realization aspect of the SOS, as well as the differences in the notions perceived by the various participants. The paper then explores the relationship between knowledge and control, and the importance of such a relationship to the SOS. In addition, the paper examines the notion of an Information Technology (IT) acquisition scheme, then demonstrates by example, a successful acquisition transition. In closing, the paper identifies available options for development optimisation and implementation of the SOS concept, and its implications for Defence and Research SOS.

Author

Stephane Collignon
Electronic Warfare Division

Stephane Collignon graduated in France in 1978 in Computer sciences. He worked for 3 years in Paris for the Burroughs (now Unisys) Detroit Program Product Division to integrate magnetic cheque high-speed sorters in the clearing-house of a commercial bank. He migrated to Australia in 1981 and worked for the Standard Chartered Bank group as system support in Sydney until 1985. He then joined Honeywell Information Systems and worked as System Support Engineer until 1990. He worked in Woomera as Mission Software support for three years with important exposure to DT&E and OT&E. Stephane was accepted for employment with Computer Sciences Corporation as Senior Member of Technical Staff in 1994 and worked mainly for the Mine Warfare project and in the Trusted Systems department. He joined DSTO in 1998 in the Weapon System Division, and is now working in the Elecronic Warfare and radar Division. Stephane is a member of the Australian Computer Society and of the Institute of the Electrical and Electronics Engineers. He was also president of the Southern Cross International Test and Evaluation Chapter from 1999 to 2001. He gained a Masters Degree in Computer Sciences at Deakin University in 1998 and is preparing a PhD. in Computer Sciences at the SEEC, University of South Australia.

Contents

1. Introduction

Today's software systems, which are built with Commercial-Off-The Shelf (COTS) hardware and software, are omnipresent. As these systems gradually replace the few remaining legacy systems, they merely substitute cost and performance problems with more complicated issues arising from the use of less controllable resources. Collignon and Cook (2002) [1] have shown that not only are COTS software products frequently utilised in environments that are completely at odds with those environments for which they were originally and specifically designed; but that they also come with limited documentation and a dearth of information about the design of the software products. This situation often leads to difficulties in the combined use of multiple COTS products in the development of a system-of-systems (SOS) architecture. Initially of this paper addresses the definition of SOS, the players in the design and realization aspect of the SOS, as well as the differences in the notions perceived by the various participants. The paper then explores the relationship between knowledge and control, and the importance of such a relationship to the SOS. In addition, the paper examines the notion of an Information Technology (IT) acquisition scheme, then demonstrates by example, a successful acquisition transition. In closing, the paper identifies available options for development, optimisation and implementation of the SOS concept, and its implications for Defence Research SOS.

2. SOS: from System to Software

Hitchins (2001) [2] gives a description of the Dynamic Systems Model (DSM) highlighting the basic relationships between components of a DSM. This DSM is also the basic representation of a SOS, defined by Cook *et al.* (1999) [3], as shown in Figure 1:

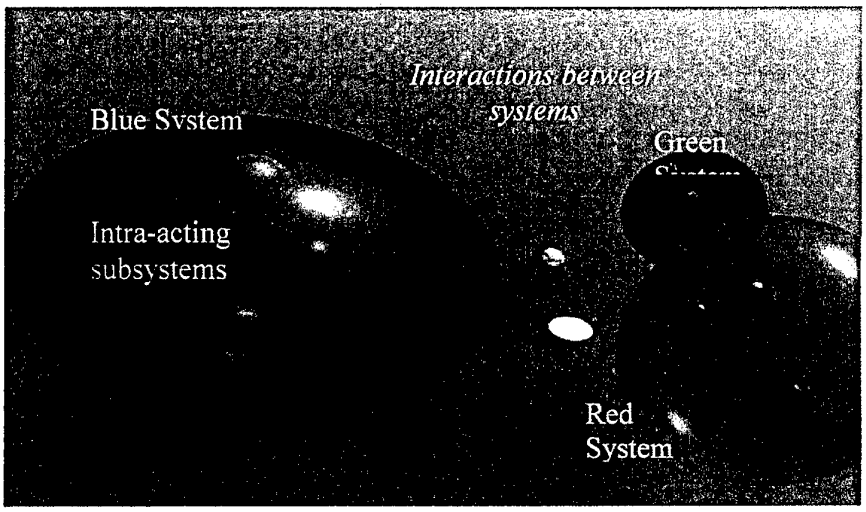


Figure 1: Sub-Systems Interactions

In Figure 1, the interactions are shown between:

- System components,
- Intra-acting subsystems within the system components, and
- The host environment.

The SOS is a result of the co-operation of all of the system components within the host operational environment. The SOS functionality and reliability are directly dependent upon the successful integration of all components into the SOS. Collignon and Cook (2002) [1] have identified and analysed the issues resulting from the integration of COTS software, originally developed for personal use, into professional system developments. These issues are particularly relevant to the design and development of SOS, which contain a large proportion of complex software systems. Table 1 highlights these issues, which are mainly software-based. The problems encountered derive from the fact that each COTS product would have been developed around a particular commercial set of views, which in all likelihood are very different from those of the SOS.

Table 1: Areas of issues per COTS characteristics

Software Attributes (Area of Issues)	Characteristics of COTS software in a Domestic Environment	Characteristics of COTS software in a Professional Environment	Alignment
Cost	Important	Secondary	Yes
Reliability	Secondary	Vital	No
Support Type	Restricted to 'How to Use'	Ability to get more than 'How to Use" is Essential	No
Duration of Support	Warranty only, at most a few years	Life-Cycle Critical	No

A dichotomy exists between the 'system engineering' and 'software engineering' approaches towards the design, modelling and construction of the SOS, the latter having to cope with an injection of extra degrees of freedom (DOF) in addition to those of the system engineering directives. These additional DOF, stated by Collignon and Cook (2002) [1] are the result of the fluctuations of the software resources used to map the system engineering design, and as such, have to be controlled to ensure successful mapping of the system engineering requirements by the software engineer management. In fact, the fluctuations brought to the products are induced by the market behaviour of the software manufacturers. Other issues that may be also considered in Table 1 include:

- Lack of in-depth documentation concerning the internal nature of the software,
- Lack of details concerning the list of compatible and non-compatible platforms with the software,
- Application Programming Interface (API) limited to the targeted market, generic API not supplied,
- Little, if any, in-built software performance monitoring – requires normally in-depth host system administration knowledge and glue code, and
- Little or no possibility of modifying the application software drivers, to design specific system interfaces.

Whilst this list of critical issues is by no means restrictive, it does, however, considerably impact upon the software developer's ability to integrate multiple COTS products capabilities. Since the early seventies, this has led to the creation of an informal, but now common, qualification known as the Software Architect (SA). It should be noted that this qualification is not derived from a formal university degree, nor does it correspond to a clear definition in computing. Despite this lack of clear role definition, many software professionals assume the title of 'SA' and so in an attempt to

define this title, it is first necessary to understand what it refers to. Shaw and Garland (1996) [4] propose this definition of Software Architecture:

"...Abstractly, Software Architecture involves the description of elements from which systems are built, interactions among those elements, patterns that guide their compositions, and constraints on these patterns..."

Having defined Software Architecture, we may now try to define the term "Architect". To do so we will use the distinction between the characteristics of the Architecting and Engineering roles, as stated by Rechtin and Maier (2000) [5]. The role of Architecting does not exactly fit the description of the engineering role. Table 2 shows how goals are perceived, methods are used and notions understood by both parties.

Table 2: the Architecting-Engineering Continuum (simplified) [5]

Characteristics	*Architecting*	Engineering
Situation/goals	Ill-structured Satisfaction	Understood Optimisation
Methods	*HEURISTICS* Synthesis Art and Science	Equation Analysis Science and Art
Interfaces	Focus on 'misfits'	Completeness
System integrity maintained through	'Single mind'	Disciplined methodology and process
Management issues	Working for the client Conceptualisation Certification Confidentiality	Working for builder Meeting Project requirements Profit versus cost

To choose a particular method, Rechtin and Maier (2000) [5] offered a basic choice of the four architecting methodologies indicated in Table 3 below:

Table 3: Architecting Methodology Types [5]

Methodology Type	Methodology Base	Applied to
Normative	Solution	Building code and communication standards
Rational	Method	System analysis and engineering
Participative or argumentative	Stakeholder	Concurrent engineering and brainstorming
Heuristics	Lessons learned	Heuristics knowledge rules

Initially, Normative and Rational methodologies were related to Science, just as Participative and Heuristics related to the Arts. In a similar fashion, the methodologies in architecting are distinctly separated, just as they are in engineering. However, the progression in technology, and in particular computing, have extended the limits of each methodology, to the extent to which there can be a partial overlapping and complementary nature between them. In this view, the heuristics methodology as proposed by Rechtin (1991) [5], and Rechtin and Maier (2000) [6], is materialised by the definition of a personal kit of heuristic tools, specific to each category of professional practitioner. Heuristics results here come from past experience in the field with proven practices. This type of heuristics methodology is partially applicable to the SA, when information on past experience with existing products is available. However Ulrich (1983) [7] approach, attempting to reach a consensus on the nature of the system to be built, in an iterative mode with the stakeholder, is also relevant when:

- SOS are built using only COTS and
- COTS knowledge, as seen above, is not available.

According to Rechtin and Maier (2000) [5] the rational methodology type can be split into sub-activities related to the tools used in the application of the methodology. An historical approach to the evolution of knowledge in the IT industry may illustrate this theory. After 1980, software, the already growing component of the systems being built, became a major component. The consequences of this extension were multiple:

- Apparition of new formal skills, such as programming methodologies, standards and languages, and
- Creation of heuristics knowledge, companion to these skills, applied to the system structure and behaviour.

With these combined skills, and the requisite knowledge, the SA is the heuristics binder between software professionals, clients and system builders, putting forward the architectural design, ahead of the engineering design. In fact, the creation and the architecture choice for a system are heavily dependent on the successful selection of the compatible commercial components that are sub-parts of the architecture.

This last statement represents only the tip of the iceberg, since a software professional will claim that Software Architecture also resides in the structure of the code written for the system. Software Architecture can therefore be said to involve descriptions of:

- Elements from which systems are built,
- Interactions between these elements,
- Patterns that guide their composition, and
- Constraints on these patterns

as stated by Shaw and Garland (1996) [4].

In a more comprehensive way, the role of the SA is described in Table 4, from BredMeyer and Malan (1999) [8]:

Table 4: Technology Competency Summary [8]

What you KNOW	What You DO	What You ARE
In-depth understanding of the domain and pertinent technologies Understand what technical issues are key to success Development methods and modelling techniques	Modeling Tradeoff analysis Prototype/experiment/simulate Prepare architectural documents and presentations Technoloy trend analysis/roadmaps Take a system viewpoint	Creative Investigative Practical/pragmatic Insightful Tolerant of ambiguity, willing to backtrack, seek multiple solutions Good at working at an abstract level

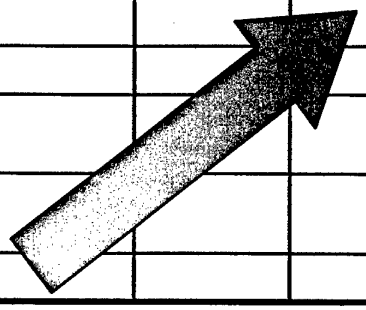
Table 4 clearly shows the magnitude of the knowledge held by the SA. Above all, the senior architect has to be seen as a ***leader*** across different areas of expertise by BredMeyer and Malan (1999) [8], such as:

- Technology
- Business strategy
- Organizational politics and
- Consulting.

The multi-skilled nature of the SA will not be expanded upon here. Table 5, as stated by BredMeyer and Malan (1999) [8] clearly shows the importance of the leader's knowledge actively reaching the project critical area:

Table 5: Software Architect Personal Characteristics [8]

	What you KNOW	What you DO	What you ARE
Leadership			
Consulting			
Organizational Politics			
Business Strategy			
Technology			



The following section, Section 3, addresses the nature of the knowledge held by the SA, followed by the way to acquire and control this knowledge.

3. Knowledge: Nature, Acquisition and Control

SA knowledge could be defined as the sum of the critical heuristic knowledge collected, while porting system engineering concepts and projects into software engineer products, during a significant time scale. On small projects, the SA may be a senior software engineer, but on medium and large-scale projects, the position is normally distinct from the software engineers. One important aspect of SA knowledge is that its limits are moving constantly, each time it is invoked.

Therefore SA knowledge may be considered as a systematic boundary critique of the system, as stated by Ulrich (2001) [9], characterised by boundary questions, to assume the progression of this knowledge. The need for SA knowledge is a consequence of the industrial evolution of IT since the 1960s. It is interesting to note that at the age of the first commercialisation of Von Neumann type machines, the manufacturer was fully in control of the delivery of the architecture, providing both hardware and software. The quick progression of technology has split this controlled link between both sides of the knowledge of the architecture, enabling a proliferation of third-party hardware and software manufacturers to deal with. The immediate consequence was that the creation of the architecture for a system would be from that time, heavily dependent on successful selection of the compatible commercial components that are parts of the architecture. This split ratio of production (hardware/software) lead to a situation where the availability of software/hardware capabilities became dependent on market rules. However this international market situation did not make every marketed product accessible to all, a situation mostly due to knowledge and product export restrictions from the software manufacturing countries.

SOS design and management solutions (Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance (C4ISR) and the like) are now available which involve a significant investment of resources, and for SOS, of considerable size. However, the multiple views system proposed by C4ISR still does not provide synchronisation with the lifecycle of the software components resources associated to the end-SOS. This in turn should cause concern, since the software components resources used might disappear or be modified during the SOS building process, thus requiring major review of the initially planned SOS development plan. This major problem could be solved by introduction of a software resource map. A software resource map, created at the time of the early SOS modelling step, should be updated when software resources are modified and checked at each milestone of the project, for any modification in the status of allocated resources. Action should be taken to resume processing in a consistent manner according to the SOS building process.

Another way of handling variations in the allocation of resources is to simply own them. This product transition from COTS to Governments-Off-The-Shelf (GOTS), involves significant resources allocation to:

- Purchase products, documentation, sources and associated development tools,
- Train and handle the maintenance staff,
- Purchase and maintain the eventual hardware components supporting the product, and,
- Draft a legal agreement to cover all of these agreements.

However, alternative solutions may be available in another context.

4. Knowledge and Control: Alternate Solution with The Japanese Experience - The Knowledge Approach

In the 1970s when Japan commenced its Information Technology (IT) industry, it did so in a climate not unlike that of Australia. Japan, however, did have the added burden of a foreign language to overcome during its introduction of new technologies. Japan IT is based on the same quality concept employed by other Japanese industry products. One may ask how Japan has assimilated foreign technology and documentation and brought it to the integrated quality level of today.

Nonaka and Nishiguchi (2001) [10] recently published a selection of papers showing their approach to optimisation of the use of knowledge. Before attempting to explain Nonaka's own theory, it is necessary to define the platform to which his theory applies. Nonaka et al (1998) [11] show the importance of total knowledge management and the associated concept of 'Ba'. The relevant definition of 'Ba' is an existential platform for interaction between individual and/or collective knowledge (Nonaka *et al.*) (2002) [12].

Knowledge is stored in everybody's 'Ba' s theoretical space. It should be mentioned at this point that knowledge in Japan is actively propagated to the 'Ba' platform where knowledge-dependent activities are conducted. The catalyst of successful interaction for knowledge exchange is the existence of specific characteristics among the knowledge owners and recipients. These characteristics are known as the "Big Five", stated by McCrae and Costa 1984) [13]:

- Extraversion
- Agreeableness
- Conscientiousness
- Emotional stability and
- Intellect

The existence of such characteristics in the personalities of the individuals taking part in the knowledge exchange clearly shows that a behavioural, though conceptual, framework exists, in support of the optimisation of knowledge exchange processes.

In contrast, the approach of the western world at large is that knowledge must be disseminated on a 'need-to-know' basis (Bush *et al.*, 2002) [14] and an open approach, as suggested in this paper is not characteristic of the western engineer. Nonaka and Nishigushi (2001) [15] stated that Knowledge is split into two complementary types:

- Explicit - that which can be expressed in words and numbers , and shared in the form of data
- Tacit - that which is difficult to pass on to others; insights, intuitions, hunches

In order to create a knowledge embedding the two types, Nonaka *et al.* (2000) [12] proposed a model aiming at:

- Facilitation of the conversion of knowledge
- Propagation of knowledge within the 'Ba' platform

This model contains four phases: Socialization, Externalisation, Combination and Internalisation (SECI), as depicted in Figure 2.

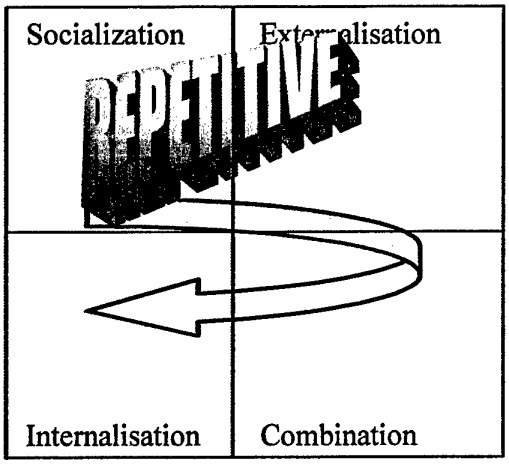


Figure 2: The SECI Process

The SECI model depicted in Figure 2 contains four sub-processes, as listed in Table 6:

Table 6: SECI sub-processes description

Sub-Process Name	Knowledge Type	Sub-process Aim
Socialization	From Tacit to Tacit	Tacit Knowledge accumulation
Externalisation	From Tacit to Explicit	Use dialogues to create concepts
Combination	From Explicit to Explicit	Formatting and transmission of new created concepts
Internalisation	From Explicit to Tacit	Share and search for new values from new concepts/visions

The Japanese approach to the diffusion of knowledge is the key to the success of their IT industry. Indeed, the IT culture was simply approached from an industry view-point and its assimilation for production of input to the rest of the Japanese industry was finally exposed in the 80s to the same constraints, as any other Japanese production unit. One of these constraints is that Knowledge is assimilated to control and promote quality in the perfect development environment, 'Ba'. Therefore the question is to define the origin of the knowledge used by Japan to create quality, industry and science integrated IT products. In this view, being able to evaluate, improve, modify, promote a software product, the originator must be able to access and modify the software source code.

There are multiple references to the use of Open Source and GNU products in the Japanese IT industry, some of them, as Thomas and Jacoby (1997) [16] in the heart of

the implementation of the quality control process itself. If we go back to the existence of these types of tacit and explicit knowledge, in COTS Software Architecture, then explicit knowledge will be restricted to the marketing views of the product, and tacit knowledge may be nonexistent, since marketing is normally done before the product is created.

It is therefore unlikely that COTS products, in their present form, have been the primary cause of successful IT integration into Japanese Industry and Research, the hypothesis being that 'Von Neumann' type systems, with uniformity between hardware and software, were actively used in a first step:

- To propagate IT knowledge to the industry,
- To elaborate an adapted SECI model for IT, and
- To plan for a transitional step leading to the creation of Japanese IT products, widely used in the Japanese Industry.

The Japanese computer manufacturer Hitachi has been one of the vectors for integrating IT knowledge in Japan in the first stage. The International Business Machines (IBM) Multiple Virtual System (MVS) was successfully imported to the Hitachi mainframes, with a limited commercial success outside Japan. This integration exercise was conducted for knowledge propagation purposes. In a similar way, the use of Open Sources Resources, as stated by Nakakoji *et al.* (2002) [17], was another factor in the Japanese success. These resources contained the knowledge attributes (visibility, use, diffusion) that are essential to the workability of the SECI model. This is, however, just a transient step to the optimisation of Information Technology research in the Japanese Industry realised on a multi-industrial scale.

5. Conclusion

The West has developed sophisticated methodologies and models to develop SOS with COTS products. These methodologies cater for large SOS and have a considerable execution time. For this reason it would be advisable to draw up a map of the planned COTS resources of the SOS. This map should be:

- Updated, when market changes impact the availability of the planned COTS resources, and
- Critically used: this means reviewing the availability of the system components - and their current suitability, compared to the initial system mapping, and restart the processing steps if necessary

However, the issues resulting from the discontinuity of the resources, compared to the current system modelling, design or build level of the SOS, have to be solved in a timely manner, to allow the continuity of the SOS building process. To expand the IT knowledge base upon which research is presently working (mainly COTS-related),

repositories of Open Sources software should be created, updated, advertised and specific tools should be adapted, developed and used by the research community. This would provide a viable alternative to faulty or missing COTS components. This will also re-inject into the Research community, some critical software development and evaluation expertise, which is gradually disappearing as a result of COTS use. This expertise would expand, when knowledge related to a particular COTS product is traditionally not retained, after delivery of the system. The Open Sources Resources should be taken advantage of in a more formal way. This solution would cater for the resolution of "integration" problems between expectations from the Research perspective, and deliveries from the Software Engineering perspective. Simultaneously, additions in University programs of a subject dedicated to the use of Open Resources would certainly re-inject the necessary knowledge for system design and modification and maintenance. This knowledge would include the evaluation and design of the host systems, as well as the interfaces to multiple applications requirements. In addition it would allow the successfully critical selection of COTS products, or alternatively, provide the ability to create a suitable software environment for the target system.

One important aspect of the successful acquisition and diffusion of knowledge, as suggested by the SECI model, is the existence of a cultural environment - 'Ba' - suitable for the knowledge exchanges. To work in an optimised manner, the presence of specific characteristics among the participants would be an important catalyst for knowledge exchange. These specific characteristics are extraversion, agreeableness, conscientiousness, emotional stability and intellect, as stated by McRae and Costa (1984) [13].

It would be reasonable to suggest that the acquired knowledge would provide a necessary critical view for the future COTS users, particularly those in Research. This critical view is essential in order to successfully achieve a design, and build of a target system, by allowing the knowledgeable selection or rejection of its basic components, wherever they belong to a challenged COTS-market, or to a repository of the available Open Sources Resources.

6. Bibliography

[1] Collignon S. and Cook S. (2002), "*Review of Software Integration and T&E Techniques in the Commercial Environment*", SETE2002, Sydney, Australia.

[2] Hitchins D.K., "*Basic Models for System Thinking*", www.Hitchins.co.uk, 2001.

[3] Cook S.C., Lawson E. and Allison J.S. (1999), "*Towards a Unified Systems Methodology for Australia Defence Systems-of-Systems*", Proceedings of INCOSE 1999 Annual Symposium, Brighton, UK, July 1999.

[4] Shaw M. and Garland D (1996), "*Software Architecture*", ISBN: 0131829672. Publisher: Prentice Hall, Upper Saddle River, N.J. ,USA.

[5] Rechtin E. and Maier (2000)."*The art of systems architecting*". ISBN: 0849304407. Publisher: CRC, Boca Raton, Fl. (USA).

[6] Rechtin E. (1991) "*System architecting: creating and building complex systems*". ISBN: 0138803455. Publisher: Prentice Hall, Englewoods Cliffs, N.J. ,USA.

[7] Ulrich W.(1983) "*Critical Heuristics of social planning*", ISBN: 0471953458, Publisher: Wiley, West Sussex, England

[8] Bredemeyer D. and Malan R. (1999), "*The Role of the Architect in Software Development*", white paper, Hewlett Packard.

[9] Ulrich W., "*The Quest for Competence in Systemic Research and Practice*", white paper, University of Lincolnshire and Humberside, Lincoln, UK. Publisher: Wiley, 001

[10] Nonaka I. and Nishigushi T. (2001)., "*Knowledge Emergence*", ISBN:0195130634. Publisher Oxford University Press, New York ,USA.

[11] Nonaka I, Reinmoeller P and Senoo D. (1998), "*Management Focus - The 'ART' of Knowledge: Systems to Capitalize on Market Knowledge.* " European Management Journal, Volume 16, No 6, pp 673-684. Publisher Elsivier Sciences, UK.

[12] Nonaka I., Reinmoller P. and Senoo D. (2002), "*Integrated IT Systems to Capitalize on Market Knowledge*", Knowledge Creation, ISBN: 0-312-22974-7, Palgrave, N.Y., USA

[13] McCrae RR and Costa PT., (1984), "*Emerging Lives, Enduring Positions: Personality in Adulthood*". Little Brown: Boston, MA ,USA.

[14] Bush P, Richards D and Dampney K., (2002), "*Visual mapping of articulable tacit knowledge*", Department of Computing Macquarie University, Sydney, Australia, 2109.

[15] Nonaka I. and Nishigushi T.,(2001).,"*Knowledge Emergence: Emergence of Ba,*", ISBN:0195130634. Publisher Oxford University Press, New York (USA).

[16] Tohma Y. and Jacoby R. (1997), "*Parameter Value Computation by Least Square Method and Evaluation of Software Availability and Reliability at Service-Operation by the Hyper-Geometric Distribution Software Reliability Growth Model*", white paper, Department of Computer Science, Tokyo Institute of Technology, Okayama 2-12-1, MeguroKu, Tokyo 152, Japan.

[17] Nakakoji K., Yamamoto Y., Nishamika Y., Kishida K., and Ye Y. (2002), "*Evolution patterns of Open Sources Software and Communities*", white paper, SRA Key Technology Laboratory Grad, School of Information Science, NAIS T3-12 Yotsuya, Shinjuku, Tokyo, 160-0004, Japan. RESTO, JST Japan Society for the Promotion of Science, 8916-5, Takayama, Ikoma, Nara, 630-0101, Japan.

Commercial-Off-The-Shelf (COTS) Systems, Architecture and Knowledge

Stephane Collignon

(DSTO-TR-1493)

AUSTRALIA

		Number of Copies
1. DEFENCE ORGANISATION		
Task Sponsor: DGMD		1
S&T Program		
Chief Defence Scientist	shared copy	1
FAS Science Policy	shared copy	1
AS Science Corporate Management	shared copy	1
Director General Science Policy Development		1
Counsellor Defence Science, London		Doc Data Sheet
Counsellor Defence Science, Washington		Doc Data Sheet
Scientific Adviser to MRDC Thailand		Doc Data Sheet
Scientific Adviser Joint		1
Navy Scientific Adviser		1
Scientific Adviser DMO ELL		Doc Data Sheet 1 x distribution list
Scientific Adviser DMO M&A		Doc Data Sheet 1 x distribution list
Scientific Adviser - Army		Doc Data Sheet 1 x distribution list
Air Force Scientific Adviser		Doc Data Sheet 1 x distribution list
Director Trials		1
Systems Sciences Laboratory		
EWSTIS (soft copy for accession to EWSTIS Web site)		1
Chief, Electronic Warfare and Radar Division		Notification
Research Leader, RF Electronic Warfare		Notification
Research Leader, EW Systems		Notification
Research Leader, EO Electronic Warfare		Notification
Research Leader, Microwave Radar		Notification
Head, ES Systems		Notification

Head, RF Countermeasures	Notification
Head, RF Technologies	Notification
Head, Strategic and Land EW	Notification
Head, Aerospace Systems	Notification
Head, Maritime Systems	Notification
Head, NavWar	Notification
Head, EO Threat Warning	Notification
Head, EO Countermeasures	Notification
Head, EO Systems & Technologies	Notification
Head, Radar Systems and Technologies	Notification
Head, Radar Modelling and Analysis	Notification
Head, Radar Signatures	Notification
Stephane Collignon,	1
DSTO Library	
Library Edinburgh	1
Australian Archives	1
Capability Systems Division	
Director General Maritime Development	Doc Data Sheet
Director General Aerospace Development	Doc Data Sheet
Director General Information Capability Development	1
Director General Simulation and Modelling (DGSIM)	Doc Data Sheet
Office of the Chief Information Officer	
Deputy CIO	Doc Data Sheet
Director General Information Policy and Plans	Doc Data Sheet
AS Information Structures and Futures	Doc Data Sheet
AS Information Architecture and Management	Doc Data Sheet
Director General Australian Defence Information Office	Doc Data Sheet
Director General Australian Defence Simulation Office	Doc Data Sheet
Strategy Group	
Director General Military Strategy	Doc Data Sheet
Director General Preparedness	Doc Data Sheet
HQAST	
SO (ASJIC)	Doc Data Sheet

Navy

SO (SCIENCE), COMAUSNAVSURFGRP, BLD 95, Garden Island, Locked Bag 12, PYRMONT NSW 2009	Doc Data Sheet 1 x distribution list
Director General Navy Capability, Performance and Plans, Navy Headquarters	1
Director General Navy Strategic Policy and Futures, Navy Headquarters	1

Army

SO (Science), Deployable Joint Force Headquarters (DJFHQ) (L), MILPO Gallipoli Barracks, Enoggera QLD 4052	Doc Data Sheet
SO (Science), Land Headquarters (LHQ), Victoria Barracks NSW	Doc Data Sheet + Executive Summary
NPOC QWG Engineer NBCD Combat Development Wing, Tobruk Barracks, Puckapunyal, 3662	Doc Data Sheet

Air Force

SO (Science) - Headquarters Air Combat Group, RAAF Base, Williamtown NSW 2314	Doc Data Sheet + Executive Summary

Intelligence Program

DGSTA, Defence Intelligence Organisation	1
Manager, Information Centre DIO	1
Assistant Secretary Corporate, Defence Imagery and Geospatial Organisation	Doc Data Sheet

Defence Materiel Organisation

Head Airborne Surveillance and Control	Doc Data Sheet
Head Aerospace Systems Division	Doc Data Sheet
Head Electronic Systems Division	Doc Data Sheet
Head Maritime Systems Division	Doc Data Sheet
Head Land Systems Division Doc Data SheetHead Industry Division	Doc Data Sheet
Chief Joint Logistics Command	Doc Data Sheet
Management Information Systems Division	Doc Data Sheet

Head Materiel Finance	Doc Data Sheet
Defence Libraries	
Library Manager, DLS-Canberra	1
Manager, DLS - Sydney West	Doc Data Sheet

OUTSIDE AUSTRALIA

INTERNATIONAL DEFENCE INFORMATION CENTRES	
US Defense Technical Information Center,	2
UK Defence Research Information Centre,	2
Canada Defence Scientific Information Service,	(pdf format) 1
NZ Defence Information Centre,	1
ABSTRACTING AND INFORMATION ORGANISATIONS	
Library, Chemical Abstracts Reference Service	1
Engineering Societies Library, US	1
Materials Information, Cambridge Scientific Abstracts, US	1
Documents Librarian, The Center for Research Libraries, US	1
SPARES	5

Total number of copies: 31

Page classification: UNCLASSIFIED

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION
DOCUMENT CONTROL DATA

1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)

2. TITLE

Commercial-Off-The-Shelf (COTS) Systems, Architecture and Knowledge

3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)

Document (U)
Title (U)
Abstract (U)

4. AUTHOR(S)

Stephane Collignon

5. CORPORATE AUTHOR

Systems Sciences Laboratory
PO Box 1500
Edinburgh SA 5111

6a. DSTO NUMBER	6b. AR NUMBER	6c. TYPE OF REPORT	7. DOCUMENT DATE
DSTO-TR-1493	AR-012-892	Technical Report	September 2003

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. NO. OF PAGES	12. NO. OF REFERENCES
E 9505-25-88	NAV 02/272	DGMD	28	17

13. URL on the World Wide Web
http://www.dsto.defence.gov.au/corporate/reports/DSTO-TR-1493.pdf

14. RELEASE AUTHORITY

Chief, Electronic Warfare and Radar Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for Public Release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED TO DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111, AUSTRALIA

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CASUAL ANNOUNCEMENT Yes

18. DEFTEST DESCRIPTORS

Commercial equipment
Systems engineering
Off the shelf equipment
Evaluation

19. ABSTRACT

Large or specialised software systems-of-systems are now mostly composed of Commercial-Off-The-Shelf (COTS) software packages. In mainstream application areas, such as office automation, this approach has proved satisfactory once the teething problems have been overcome. Defence and Research have also followed this trend in a desire to cut costs and take advantage of the functionality of commercial software. However, the difference between the often demanding requirements for defence systems-of-systems, and those against which COTS software has been designed, has resulted in unfavourable outcomes. This paper discusses the insights that can be gleaned from systems theory and practice to improve the probability of successfully integrating COTS components into defence systems. In particular the paper will provide a practical definition of systems of systems (SOS) and then will compare the generic design drivers for software of this scale with that of modest stand alone packages. A systems approach is invoked to help understand the problem context presented by the SOS evolution task and uses this to identify ideas that may be able to mitigate some of the issues. The paper will conclude by identifying some practical approaches that can help realise evolving Defence and Research SOS.